МИНИСТЕРСТВО НАУКИ И ВЫСШЕГО ОБРАЗОВАНИЯ  
РОССИЙСКОЙ ФЕДЕРАЦИИ

ФЕДЕРАЛЬНОЕ ГОСУДАРСТВЕННОЕ БЮДЖЕТНОЕ ОБРАЗОВАТЕЛЬНОЕ УЧРЕЖДЕНИЕ ВЫСШЕГО ОБРАЗОВАНИЯ

«ВОРОНЕЖСКИЙ ГОСУДАРСТВЕННЫЙ ТЕХНИЧЕСКИЙ УНИВЕРСИТЕТ»  
(ФГБОУ ВО «ВГТУ», ВГТУ)

Факультет информационных технологий и компьютерной безопасности

Кафедра компьютерных интеллектуальных технологий проектирования

ОТЧЁТ ПО ЛАБОРАТОРНОЙ РАБОТЕ №8

По дисциплине: «Системы хранения и обработки данных»

Тема: «Разработка физической структуры базы данных»

Выполнил работу студент группы мИИВТ-251: Вытовтова Н.Ю.

подпись, дата

Воронеж 2025

Содержание

1. Введение………………………………………………………………………………...3

2. Порядок выполнения учебной задачи.……………………………..…………………3

2.1 Основы работы с СУБД PostgreSQL…………………………………..…..…..3

2.2 PgAdmin как инструментарий для работы с СУБД.………………..………...3

2.3 SQL-запросы для создания таблиц в СУБД Postgres.......................................4

2.4 Описание SQL-запросов...……………………………………………………..6

2.5 Создание docker-контейнера для базы данных ………………………………7

2.6 Построение графической диаграммы физической структуры базы данных и ее описание …………………………………………………………………………8

3. Заключение………………………………………………………………………….....10

4 Ответы на контрольные вопросы…………………………………………………......10

Список используемой литературы……………………………………………………...16

**1 Введение**

Целью данной лабораторной работы является изучение и освоение процесса разработки физической структуры базы данных с использованием системы управления базами данных Postgres.

Задачей лабораторной работы является выполнение следующих действий:

* создание таблиц в СУБД Postgres с помощью SQL-запросов с атрибутами, связями, первичными и внешними ключами в соответствии с разработанной в предыдущей лабораторной работе логической структурой базы данных;
* построение графической диаграммы для отображения физической структуры базы данных.

**2 Порядок выполнения учебной задачи**

2.1 Основы работы с СУБД PostgreSQL

PostgreSQL (часто сокращают до «Postgres») — это объектно-реляционная система управления базами данных (ОРСУБД). Это значит, что она поддерживает не только стандартные реляционные модели (таблицы, строки, столбцы), но и объектно-ориентированные возможности (наследование, пользовательские типы данных).

2.2 PgAdmin как инструментарий для работы с СУБД

pgAdmin — инструмент с графическим интерфейсом (GUI) для управления базами данных PostgreSQL. Это веб-приложение, которое позволяет взаимодействовать с базами данных через браузер или отдельное десктопное приложение (Рис. 1).

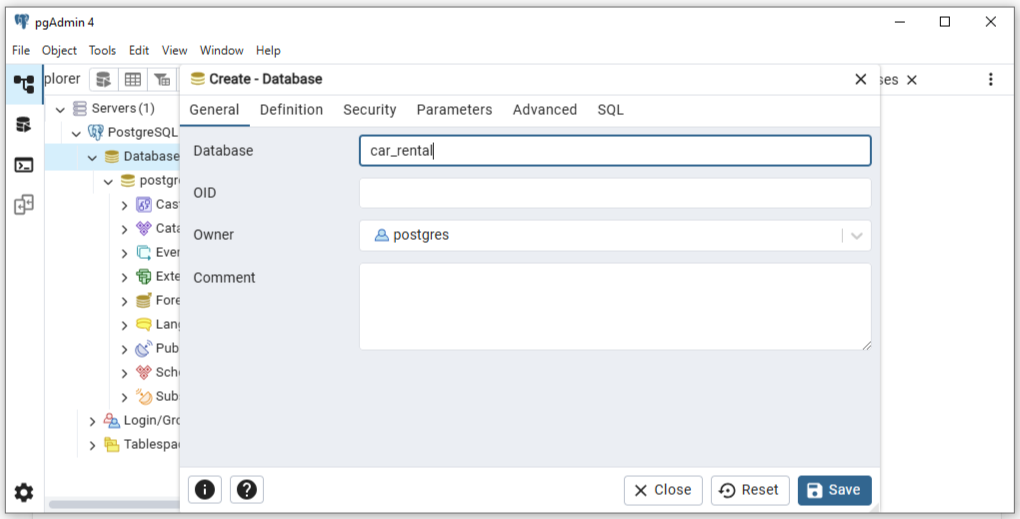


Рис. 1 – Интерфейс pgAdmin

2.3 SQL-запросы для создания таблиц в СУБД Postgres

Для создания таблиц базы данных «Прокат автомобилей» будем использовать следующие SQL-запросы:

1. Таблица CLIENTS (Клиенты)

CREATE TABLE clients (

client\_id SERIAL PRIMARY KEY,

last\_name VARCHAR(50) NOT NULL,

first\_name VARCHAR(50) NOT NULL,

patronymic VARCHAR(50),

passport\_serial VARCHAR(4) NOT NULL,

passport\_number VARCHAR(6) NOT NULL,

CONSTRAINT unique\_passport UNIQUE (passport\_serial, passport\_number)

);

2. Таблица CARS (Автомобили)

CREATE TABLE cars (

car\_id SERIAL PRIMARY KEY,

model VARCHAR(100) NOT NULL,

color VARCHAR(30) NOT NULL,

manufacture\_year INTEGER NOT NULL CHECK (manufacture\_year > 1900 AND manufacture\_year <= EXTRACT(YEAR FROM CURRENT\_DATE)),

license\_plate VARCHAR(15) NOT NULL UNIQUE,

rental\_cost\_per\_day DECIMAL(10,2) NOT NULL CHECK (rental\_cost\_per\_day > 0)

);

3. Таблица INSURANCE\_POLICIES (Страховые полисы)

CREATE TABLE insurance\_policies (

policy\_id SERIAL PRIMARY KEY,

car\_id INTEGER NOT NULL,

insurance\_value DECIMAL(12,2) NOT NULL CHECK (insurance\_value > 0),

start\_date DATE NOT NULL,

end\_date DATE NOT NULL,

CONSTRAINT fk\_car FOREIGN KEY (car\_id) REFERENCES cars(car\_id) ON DELETE CASCADE,

CONSTRAINT valid\_dates CHECK (start\_date < end\_date)

);

4. Таблица RENTALS (Прокаты)

CREATE TABLE rentals (

rental\_id SERIAL PRIMARY KEY,

client\_id INTEGER NOT NULL,

car\_id INTEGER NOT NULL,

start\_date DATE NOT NULL,

rental\_days INTEGER NOT NULL CHECK (rental\_days > 0),

total\_cost DECIMAL(10,2) NOT NULL CHECK (total\_cost > 0),

CONSTRAINT fk\_client FOREIGN KEY (client\_id) REFERENCES clients(client\_id) ON DELETE CASCADE,

CONSTRAINT fk\_car\_rental FOREIGN KEY (car\_id) REFERENCES cars(car\_id) ON DELETE CASCADE

);

2.4 Описание SQL-запросов

1. CREATE TABLE - основной оператор для создания таблиц

SERIAL - автоматически инкрементируемый целочисленный тип (аналог AUTO\_INCREMENT)

PRIMARY KEY - определение первичного ключа

VARCHAR(n) - строковый тип переменной длины с ограничением

NOT NULL - ограничение, запрещающее NULL значения

CHECK - проверочное ограничение для валидации данных

UNIQUE - обеспечение уникальности значений

CONSTRAINT - именованные ограничения целостности

2. FOREIGN KEY - определение внешнего ключа

REFERENCES - указание на связанную таблицу и поле

ON DELETE CASCADE - автоматическое удаление зависимых записей

3. COMMENT - добавление комментариев к таблицам и столбцам

Улучшает документацию базы данных

4. CREATE INDEX - создание индексов для ускорения запросов

Ускоряет поиск по часто используемым полям

2.5 Создание docker-контейнера для базы данных

Создаем файл docker-compose.yml со следующим содержимым:

version: '3.8'

services:

postgres:

image: postgres:15-alpine

container\_name: car\_rental\_db

environment:

POSTGRES\_DB: car\_rental

POSTGRES\_USER: admin

POSTGRES\_PASSWORD: password123

ports:

- "5432:5432"

volumes:

- postgres\_data:/var/lib/postgresql/data

- ./init.sql:/docker-entrypoint-initdb.d/init.sql

restart: unless-stopped

pgadmin:

image: dpage/pgadmin4

container\_name: car\_rental\_pgadmin

environment:

PGADMIN\_DEFAULT\_EMAIL: admin@car-rental.com

PGADMIN\_DEFAULT\_PASSWORD: admin123

ports:

- "5050:80"

depends\_on:

- postgres

restart: unless-stopped

volumes:

postgres\_data:

2.6 Построение графической диаграммы физической  
структуры базы данных и ее описание

После выполнения SQL-запросов физическая структура будет соответствовать созданной ERD диаграмме. Все таблицы созданы с указанными полями, типами данных, ограничениями и связями (Рис. 2).

Особенности физической реализации:

* Все первичные ключи используют тип SERIAL для автоинкремента
* Для строковых данных использован VARCHAR с разумными ограничениями длины
* Для денежных значений - DECIMAL с указанием точности
* Добавлены проверочные ограничения (CHECK) для валидации данных
* Созданы индексы для часто используемых полей
* Реализованы внешние ключи с каскадным удалением

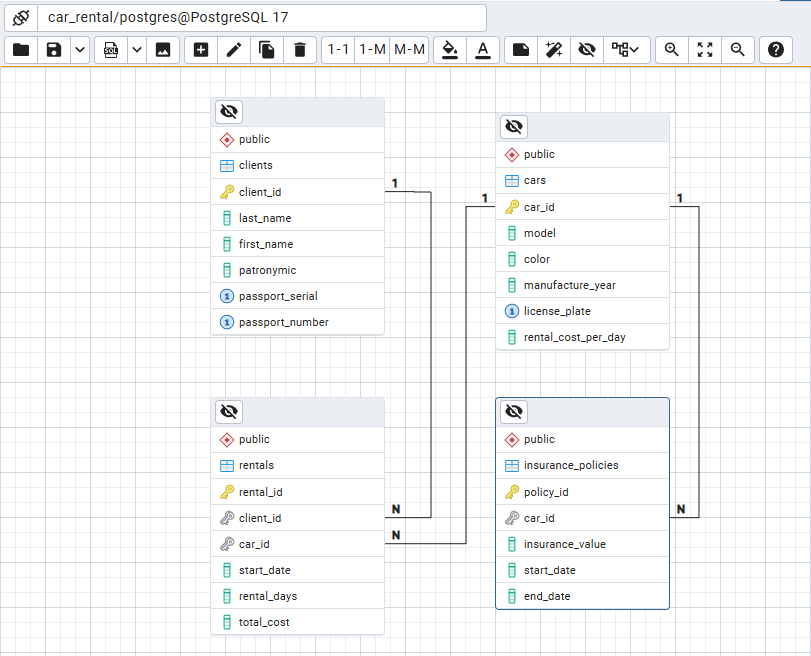


Рис. 2 – Диаграмма базы данных «Прокат автомобилей»

Построенная в ходе лабораторной работы диаграмма физической структуры базы данных наглядно отображает логическую структуру базы данных «Прокат автомобилей».

На диаграмме удалось отобразить:

1. Сущности: Clients, Cars, Rentals, InsurancePolicies.

2. Ключи: Для каждой сущности определен первичный ключ PK (например, client\_id). Внешние ключи FK (например, client\_id в таблице Rentals) четко показаны и связаны с соответствующими первичными ключами.

3. Связи:

- Clients 1 ──< Rentals («Один клиент может иметь много прокатов»).

- Cars 1 ──< Rentals («Один автомобиль может быть во многих прокатах»).

- Cars 1 ──< InsurancePolicies («Один автомобиль может иметь много страховых полисов»).

**3 Заключение**

В ходе выполнения данной лабораторной работы были успешно достигнуты все поставленные учебные задачи и приобретены практические навыки работы с таким средством для работы с СУБД как pgAdmin.

Построенная в ходе лабораторной работы диаграмма наглядно отображает физическую структуру базы данных «Прокат автомобилей» (Рис. 2).

На диаграмме удалось отобразить:

Сущности (Clients, Cars, Rentals, InsurancePolicies), ключи PK (например, client\_id) и внешние ключи FK (например, client\_id в таблице Rentals), а также связи.

Таким образом, pgAdmin – это продвинутый и интуитивно понятный инструмент с графическим интерфейсом для управления базами данных PostgreSQL. Он позволяет управлять серверами, выполнять SQL-запросы, визуализировать структуры данных и следить за производительностью базы.

**4 Ответы на контрольные вопросы**

1) Особенности PostgreSQL

PostgreSQL – это мощная объектно-реляционная СУБД с открытым исходным кодом. Основные особенности:

* Поддержка стандартов SQL: Полная поддержка SQL стандартов
* Расширяемость: Возможность создавать собственные типы данных, функции, операторы
* Надежность: ACID-совместимость, транзакции, точки сохранения
* Производительность: Сложные запросы, индексы различных типов (B-tree, Hash, GiST, SP-GiST, GIN)
* Безопасность: Ролевая модель доступа, SSL, шифрование
* Поддержка JSON: Работа с документами NoSQL внутри реляционной БД
* Географические данные: Расширение PostGIS для работы с геоданными

2) Что такое DDL?

DDL (Data Definition Language) – язык определения данных. Команды для работы со структурой БД:

CREATE - создание объектов (баз, таблиц, индексов)

ALTER - изменение структуры объектов

DROP - удаление объектов

TRUNCATE - очистка таблицы (удаление всех данных)

COMMENT - добавление комментариев

Пример: CREATE TABLE clients (...);

3) Что такое DML?

DML (Data Manipulation Language) - язык манипулирования данными. Команды для работы с данными:

SELECT - выборка данных

INSERT - вставка новых записей

UPDATE - обновление существующих записей

DELETE - удаление записей

MERGE - объединение операций вставки/обновления

Пример: INSERT INTO clients VALUES (...);

4) Что такое DCL?

DCL (Data Control Language) - язык управления данными. Команды для управления доступом:

GRANT - предоставление прав доступа

REVOKE - отзыв прав доступа

COMMIT - подтверждение транзакции

ROLLBACK - откат транзакции

Пример: GRANT SELECT ON clients TO user1;

5) С какой команды, как правило, начинается запрос?

Запрос обычно начинается с команды SELECT (для выборки данных) или с команды манипулирования данными (INSERT, UPDATE, DELETE).

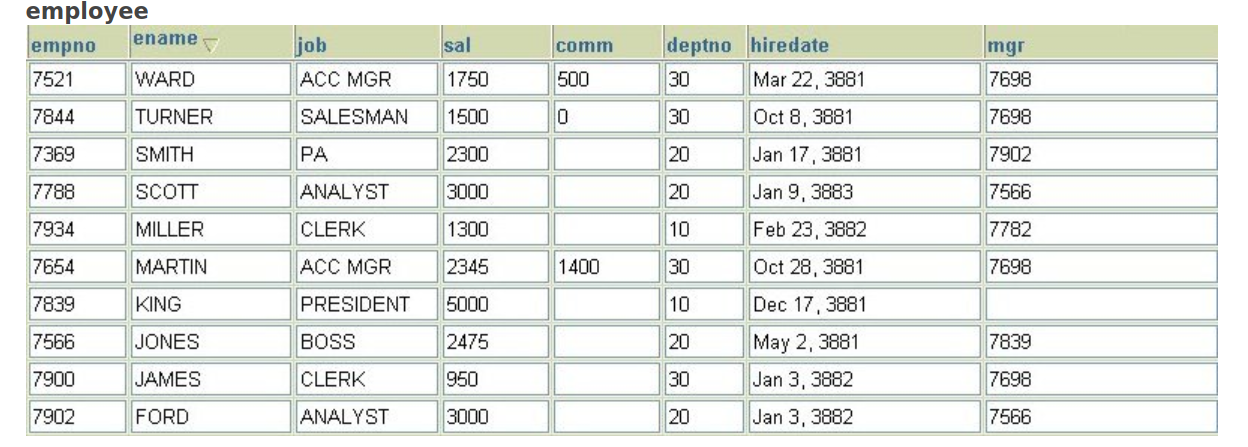
6) Какая команда используется для создания базы данных?

Команда CREATE DATABASE.

7) Какая команда используется для создания таблицы в базе данных?

Команда CREATE TABLE.

8) Существует следующая таблица:



Из таблицы employee необходимо показать сотрудников с empno 7844, 7788 и 7902 одним запросом. Какие из приведенных ниже запросов соответствуют данному условию?

• SELECT \* FROM employee WHERE mod(mod(sal, 23), 5) = 0;

• SELECT \* FROM employee WHERE mod(sal, 23) = 0;

• SELECT \* FROM employee WHERE sal = 3000;

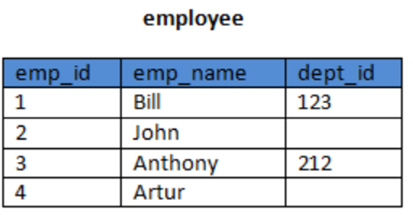
• SELECT \* FROM employee WHERE sal = 1500;

• SELECT \* FROM employee WHERE sal IN (1500, 3000);

Ответ: SELECT \* FROM employee WHERE sal IN (1500, 3000);

Так как 7844 (1500), 7788 (3000), 7902 (3000) - все три нужных сотрудника имеют sal 1500 и 3000. Других с sal 1500 или 3000 в таблице нет. Подходит.

9) В базе данных компании хранятся данные о работниках и подразделениях, к которым они относятся.



Ваша задача: достать весь список работников, при этом все работники без подразделения должны быть в начале списка. Какой из представленных запросов выполнит эту задачу?

• SELECT \* FROM employee ORDER BY dept\_id NULLS LAST;

• SELECT \* FROM employee ORDER BY dept\_id ASC;

• SELECT \* FROM employee ORDER BY dept\_id NULLS FIRST;

• SELECT \* FROM employee ORDER BY dept\_id;

• SELECT \* FROM employee ORDER BY dept\_id NULLS FIRST ASC;

Ответ:

* SELECT \* FROM employee ORDER BY dept\_id NULLS FIRST;
* SELECT \* FROM employee ORDER BY dept\_id NULLS FIRST ASC;

Оба дадут порядок:

John (NULL)

Artur (NULL)

Bill (123)

Anthony (212)

10) Какие из следующих значений подходят для типа char в PostgreSQL?

• 12

• '1'

• 1.22

• 'a'

• 'char'

Ответ: Тип char(n) (или character(n)) в PostgreSQL предназначен для символьных строк фиксированной длины. Правильные ответы: '1' - строковый литерал, 'a' - строковый литерал, 'char' - строковый литерал.

11) Какие из перечисленных конструкций запроса НЕ являются верными? Выбрать можно несколько вариантов ответа.

• Select список полей from список таблиц where условия order by поля для сортировки

• Select список полей from список таблиц union Select список полей from список таблиц order by поля для сортировки

• Select список полей from список таблиц having условия группировки group by поля для группировки

• Select список полей from список таблиц where условия group by поля для группировки order by поля для сортировки

• Select список полей from список таблиц order by поля для сортировки union Select список полей from список таблиц order by поля для сортировки

Ответ: Неправильные конструкции:

- Select список полей from список таблиц having условия группировки group by поля для группировки.

Ошибка: Неверный порядок - сначала должен быть GROUP BY, потом HAVING

- Select список полей from список таблиц order by поля для сортировки union Select список полей from список таблиц order by поля для сортировки

Ошибка: ORDER BY нельзя использовать внутри UNION перед объединением

12) Необходимо вычислить средний объем продаж (advance) книг и годовой доход (sales) от продажи книг по всем категориям (type) из таблицы titles. Какой из перечисленных фрагментов кода позволит решить поставленную задачу?

• Select avg (advance), sum (sales) from titles where type between ’business’ or ’internet’

• Select avg (advance), sum (sales) from titles

• Select avg (advance), sum (sales) from titles where type between ’business’ and ’internet’

• Select avg (advance), sum (sales) from titles group by type

• Select avg (advance), sum (sales) from titles where type = ’business’ and type = ’internet’

Правильный ответ: Select avg(advance), sum(sales) from titles, потому что этот запрос вычисляет среднее значение advance и сумму sales по всей таблице, что соответствует условию «по всем категориям».

Список используемой литературы

1. ГОСТ 7.32-2017 Система стандартов по информации, библиотечному и издательскому делу. Отчет о научно-исследовательской работе. Структура и правила оформления

2. Грас Д. 2021 Data Science Наука о данных с нуля: Пер. с англ., 2 е изд. перераб., 416 с., СПб БХВ Петербург

3. Коннолли, Т. Базы данных: проектирование, реализация и сопровождение. Теория и практика / Т. Коннолли, К. Бегг. — 3-е изд. — М. : Вильямс, 2003. — 1436 с. — ISBN 5-8459-0527-1.

4. Кузьмин, А. В. Введение в базы данных : учебное пособие / А. В. Кузьмин, И. А. Фомин. — СПб. : СПбГУАП, 2017. — 124 с. — ISBN 978-5-8088-1196-7.

5. Тарасов, С. В. Базы данных. Язык SQL : учебное пособие / С. В. Тарасов. — М. : СОЛОН-ПРЕСС, 2020. — 176 с. — ISBN 978-5-91359-365-4.